Looping over records in FileMaker with error checking

Recently I got a client complaining about a crash in FileMaker Server with a script using plugin functions. A look on the trace output from our plugin via MBS("Trace"; $path) function call to write a log file showed the problem. The loops where running endless. And an endless loop eventually causes trouble when some resource is limited, e.g. available memory.

Looking on the script we basically found this a couple of times:

```
Go to Layout [ “MyTable” (MyTABLE) ; Animation: None ]
Perform Find [ Restore ]
Go to Record/Request/Page [ First ]
Loop
	# process record
	#
	# next
	Go to Record/Request/Page [ Next ; Exit after last: On ]
End Loop
```

For privacy reasons, I removed the code to process the record and renamed the layout and file here. But otherwise the script contains a few of those loops and not all loops work well. Does the script look fine? Look and decide before you continue to read.

Adding error handling

First thing missing in the script is error handling. Everything could go wrong. First the call to "Go to Layout" may go wrong, so we need to add error handling there. Go to layout may fail for a lot of reasons. First someone may have deleted the target layout or the script step was pasted with Copy & Paste and nobody reassigned a layout. If you go to a layout by name, someone may have renamed it so the script step fails to find it. Security may kick in and disallow you leaving current layout or entering the target layout.

Next Perform Find may go wrong. Whether it's security forbidding something, missing or invalid search criteria. So let us prepare for an error here. But most importing, you may just not find records, so you should have an IF to check whether Get(FoundCount) > 0, before you enter the loop.

The next line to go to first record may fail if there are no records, if you can't leave current record or you can't enter the first record as it may be locked. Same for the Go to Next Record script step in the loop. It will exit automatically when there are no more records, but what does it do, if it can't move to the next record? Probably stay and you have the endless loop.

So let us add error checks and the script gets much longer:

```
# avoid error dialogs to user and don't let them cancel it
Set Error Capture [ On ]
Allow User Abort [ Off ]
#
# go to first step
Go to Layout [ “MyTable” (MyTABLE) ; Animation: None ]
Set Variable [ $error ; Value: Get(LastError) ]
If [ $error ≠ 0 ]
	# report to user
	Show Custom Dialog [ "Problem in script " & Get(ScriptName); "Failed to go to layout with error " & $error ]
	# return error to caller
	Exit Script [ Text Result: "Failed to go to layout with error " & $error ]
End If
#
Perform Find [ Restore ]
Set Variable [ $error ; Value: Get(LastError) ]
If [ $error ≠ 0 ]
	# report to user
	Show Custom Dialog [ "Problem in script " & Get(ScriptName); "Failed to perform find with error " & $error ]
	# return error to caller
	Exit Script [ Text Result: "Failed to perform find with error " & $error ]
End If
#
Go to Record/Request/Page [ First ]
Set Variable [ $error ; Value: Get(LastError) ]
If [ $error ≠ 0 ]
	# report to user
	Show Custom Dialog [ "Problem in script " & Get(ScriptName); "Failed to go to first record with error " & $error ]
	# return error to caller
	Exit Script [ Text Result: "Failed to go to first record with error " & $error ]
End If
#
Set Variable [ $$list ; Value: "" ]
If [ Get(FoundCount) > 0 ]
```

```
        Loop
                # process record
                #
                Set Variable [ $$list ; Value: $$List & Contacts::First & ¶ ]
                #
                # next
                Go to Record/Request/Page [ Next ; Exit after last: On ]
                Set Variable [ $error ; Value: Get(LastError) ]
                If [ $error ≠ 0 ]
                        # report to user
                        Show Custom Dialog [ "Problem in script " & Get(ScriptName);
"Failed to go to next record with error " & $error ]
                        # return error to caller
                        Exit Script [ Text Result: "Failed to go to next record with error "
& $error ]
                End If
        End Loop
End If
#
Exit Script [ Text Result: "OK" ]
```

You see we have a lot of error handling now. Can we make this shorter and log errors better? Yes, we can use FMSQL functions with MBS FileMaker Plugin to write an error log in the database. Please create a table ErrorLog and add a few fields:

TS	Timestamp
ErrorMessage	Text
ErrorCode	Number
ScriptName	Text
AccountName	Text
RecordID	Number
TableName	Text

You can add more fields as needed later. Now we define a new custom function named CheckError with a parameter Operation. The function is defined with one Let statement like this:

```
Let ( [
errorCode = Get(LastError);
errorMessage = get(LastExternalErrorDetail);
hasError = (errorCode ≠ 0);
$$error = If ( hasError ; "Failed to " & Operation & " with error " & errorCode; "");

log = If ( hasError ;
```

```
        MBS( "FM.InsertRecord"; Get(FileName); "ErrorLog";
                "TS"; get(CurrentTimestamp);
                "ErrorMessage"; errorMessage;
                "ErrorCode"; errorCode;
                "ScriptName"; get(ScriptName);
                "AccountName"; get(AccountName);
                "RecordID"; get(RecordID);
                "TableName"; get(LayoutTableName);
                "Operation"; Operation
 ))

]; hasError )
```

As you see we query error status from FileMaker and check if the error code is not zero. If we have an error, we use SQL functions in MBS FileMaker Plugin to log the error to a table. Please make sure you have a table occurrence for ErrorLog in your relationship graph.

The Operation parameter to the CheckError function is used to log what we tried to do in the script and is used to populate the error message. If you have multiple "Go To Layout" script steps, you may prefer to include a prefix to specify which calls. If ever FileMaker gets a Get(LineNumber) function, we could query it here to return the line number in the current script.

We set $$error with the error message as we return that from the script in case of an error. This helps to avoid us typing the error message twice when using the Custom Function as we can return $$error in the Exit Script step. Also you can see it in data viewer when stepping through the script.

The MBS call to FM.InsertRecord creates the log record. This is a thin wrapper around FM.ExecuteFileSQL where the plugin builds the SQL itself. Be sure to test this once to make sure you have all data types correctly. e.g. if you created RecordID as text field, you may see "[MBS] ERROR: FQL0013/(1:134): Incompatible types in assignment." as the record number from Get(RecordID) is a number. Be sure to verify the custom function works.

If you like to see a message box, you can add this line to the Let statement's list of assignments:

```
r = if (hasError; MBS("MsgBox"; $$error));
```

This uses our little Msgbox function to show a dialog box with the error message.

Check for context

Not sure if you know it, but be prepared for every script to be triggered at anytime by anyone. You probably enable fmp:// URL script triggering, AppleScript access or you may use a plugin with a start script function. Even the FileMaker iOS SDK provides a function to trigger scripts from your delegate in Swift or Objective-C.

As a little check, we may just check if the current privilege name is [Full Access] to make sure it's not a regular user account:

```
If [ Get(CurrentPrivilegeSetName) ≠ "[Full Access]" ]
	Exit Script [ Text Result: "Not allowed to run here." ]
End If
```

Some scripts are supposed to run on server or client, so you can check the server status via Get(ApplicationVersion) and exit the script if called on the wrong process:

```
If [ Position(Get(ApplicationVersion);"server";1;1) = 0 ]
	# not on server
	Exit Script [ Text Result: "Must run on server." ]
End If
```

Final Script

The final script looks like this:

```
# avoid error dialogs to user and don't let them cancel it
Set Error Capture [ On ]
Allow User Abort [ Off ]
#
If [ Get(CurrentPrivilegeSetName) ≠ "[Full Access]" ]
	Exit Script [ Text Result: "Not allowed to run here." ]
End If
#
# go to first step
Go to Layout [ “MyTable” (MyTABLE) ; Animation: None ]
If [ CheckError("Go To Layout") ]
	Exit Script [ Text Result: $$error ]
End If
#
Perform Find [ Restore ]
If [ CheckError("Perform Find") ]
	Exit Script [ Text Result: $$error ]
End If
```

```
#
Go to Record/Request/Page [ First ]
If [ CheckError("Go To First Record") ]
	Exit Script [ Text Result: $$error ]
End If
#
If [ Get(FoundCount) > 0 ]
	Loop
		# process record
		#
		# next
		Go to Record/Request/Page [ Next ; Exit after last: On ]
		If [ CheckError("Go Next Record") ]
			Exit Script [ Text Result: $$error ]
		End If
	End Loop
End If
#
Exit Script [ Text Result: "OK" ]
```

Please don't hesitate to send us your comments and questions. I wonder if we missed something?